Perspectives on Distributed Computing: August 2007 Quarterly Report

August 29, 2007
Lisa Childers, Lee Liming

Report Contents

1 User Perspectives Project Overview

The User Perspectives team has been interviewing individual members of collaborative science projects who use distributed computing technology. As part of the Community Driven Improvement of Globus Software (CDIGS), the purpose of this work is to document the work-related goals, methods, and challenges facing today's scientific technology users and to record their perspectives on Globus software and its relevance to their work.

The primary deliverable of the project will be a final paper describing the findings from the series of interviews (see Fig. 1). The project is generating a number of immediate benefits along the way. This interim report, *Perspectives on Distributed Computing: August 2007 Quarterly Report*, provides an early look at the project's accomplishments.

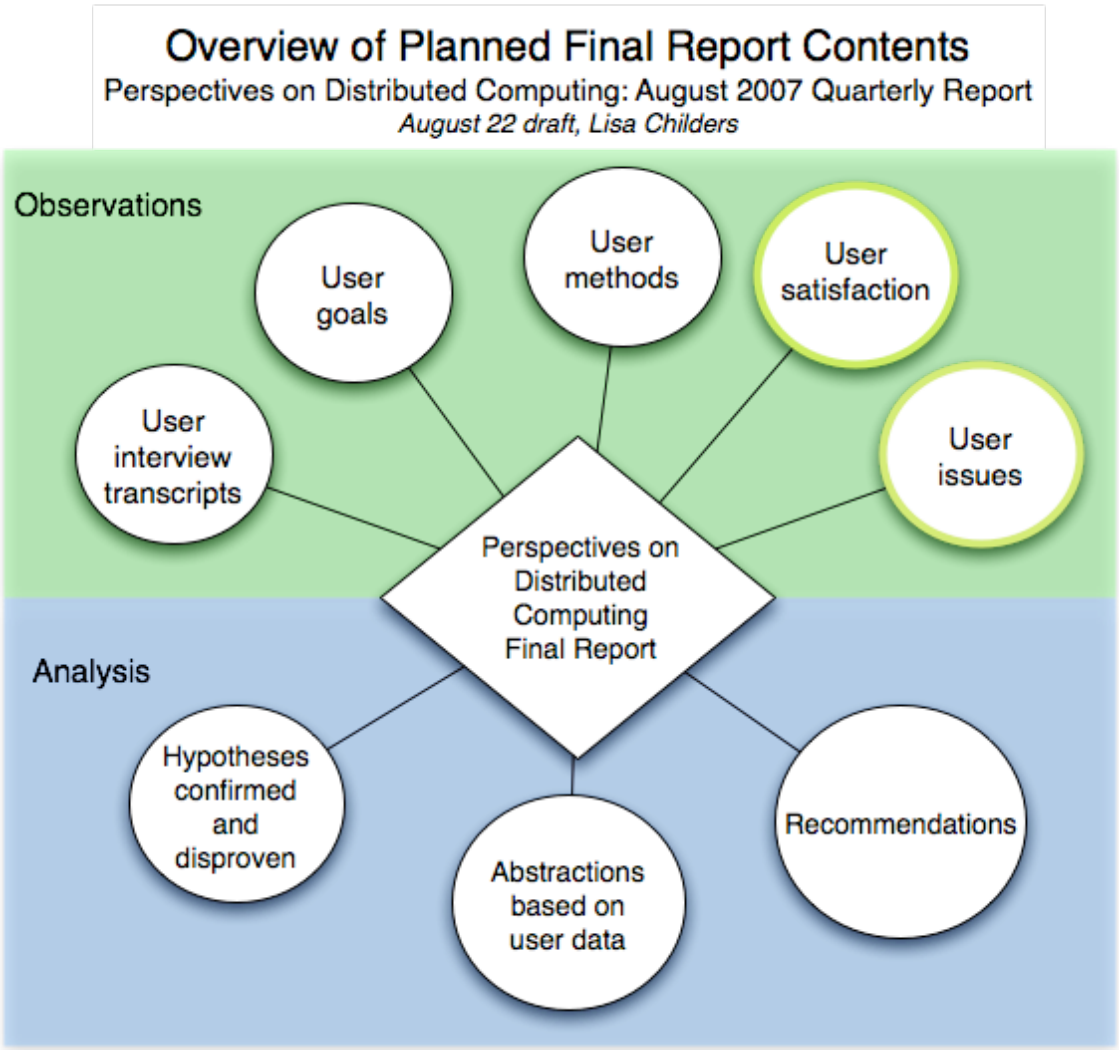

Figure 1: Planned final report contents for the User Perspectives project

At this early stage, the report mainly includes observations of user problems and expressions of user satisfaction extracted from the first interviews. The interview transcripts contain information on many additional topics. Subsequent reports will include an increasingly rich set of observations and, eventually, analytical material. Although we hope the reader will find this first report of use, its preliminary nature should be kept in mind. In particular, it would be premature to make decisions or to assign relative weights to any of the observations noted here, as we have interviewed only a small number of people to date.

The User Perspectives project is the cornerstone of the CDIGS User Advocacy program. An overview of that program, including a brief description of the work areas and recent accomplishments, can be found in the Appendix.

2 Summary of Progress to Date

During this quarter, we spent a significant amount of effort laying the foundation for the study. Having laid the foundation (and where possible in parallel with that work), we interviewed thirteen users from eleven U.S. projects spanning 24 scientific fields. Nine of the interviews were transcribed during the reporting period. Based on the results, we created an initial summary of users, user issues, and user expressions of satisfaction. The data is beginning to substantiate several issues of which we were aware, for example, a need for faster GRAM file staging and challenges regarding reliability. We also have gained some insight into problems of which we (and perhaps our team) were unaware. Moreover, the interviews are eliciting new information about the specific roles that our software plays in our users' science enterprises.

While the primary product of this activity will be a report containing the interview transcripts and an analysis of the data, the interviews have also been producing shorter-term benefits. Some of the interviews appear to validate prioritization choices made by the CDIGS team, and we hope to use these observations in the upcoming CDIGS project review. We have dispatched several issues to team leaders for follow-up. Preliminary analysis appears to explain some of the behaviors we've seen from users and points to areas where Globus can have greater impact in the community.

Several of those interviewed expressed gratitude for the opportunity to comment on our work and seem pleased that our team is actively seeking their views. Several people noted that Globus is important to their work and to their colleagues' work.

Overall, we are pleased with the quality and types of data being generated by this activity. We are confident that the results will provide the CDIGS team with unprecedented information about our users, their work, and opportunities for Globus to have a greater positive impact.

2.1 Quarterly Accomplishment #1: Study Designed

The purpose of the User Perspectives study is to paint a picture of the Globus user community (and the potential user community), providing information about the many types of users in the community, their goals, the relevance they see for Globus software in their work, and their experiences to date with using Globus software. We are not attempting to confirm or refute any particular hypotheses. We instead hope to illuminate commonalities and patterns that might be worth further, more focused, investigations, ultimately leading to awareness and follow-up on specific issues and opportunities.

2.1.1 Methodology and Format

Initially, Lisa reviewed literature on how to conduct studies of this type. Based on her review, she decided to conduct interviews following a predefined script that is the same for everyone, with a mixture of closed and open-ended questions intended to generate a mixture of structured data (e.g., demographics) and unstructured data (e.g., descriptions

of goals). The interview script encourages interviewees to describe how they use distributed computing tools for their work.

Lee shared the interview script with Ann Zimmerman at the University of Michigan and asked for her general review of the study plan. (Ann has performed a series of NSF-sponsored user studies in cyberinfrastructure areas.) Following a review of the script and a discussion with Lee, Ann indicated that she expected that the script would do a reasonable job of collecting the kinds of information needed to serve the goals of the study.

Lisa conducts all of the interviews over the phone and records the interview. She follows the interview script carefully, relying on optional "prompts" if the interviewee has misunderstood a question. Lee observes a subset of interviews (with the interviewee's permission) by sharing the phone line but remaining silent.

Our current plan is to monitor the number of new (unique) "observations" that Lisa records during transcription of each interview, and when we see that number drop substantially—that is, the interviews are no longer producing many new issues, topics, use patterns, and so forth—we will verify that we have not missed key project domains or job types and then assume that we have reached a representative sampling. (If we do not see the number of interesting points from the interviews drop after a long time, we may need to revisit this strategy.)

2.1.2 User Selection Process

We developed an interviewee selection process designed to span a broad set of science disciplines, use modes, and levels and types of engagement. After drafting an initial list of candidates through ad hoc means, we categorized the candidates using a number of criteria—project, funding agencies, science discipline, role in the project, type of project—and then reviewed the distributions across these categories to spot gaps. We continue to expand the candidate list, looking for areas that need greater representation and then brainstorming for matching candidates. We also plan to explore more objective sources of candidates such as participant lists for meetings and conferences, mailing list participation, and NSF project award data.

2.1.3 Transcription and Indexing

Following each interview, Lisa reviews the recording and produces a transcript. The transcript records the interviewee's responses to the interview questions as close to verbatim as possible. (Some editing was done to ensure readability.) While producing the transcript, Lisa adds entries to an index, recording references to specific Globus software tools, problem areas, and science issues.

2.1.4 Deferred Analysis

We are intentionally deferring a wide range of possible analysis activities until the analysis phase of the study begins. Meanwhile, we are maintaining a set of observations and candidate hypotheses on a shared (private) Wiki. These are ideas for issues that we suspect the data may be showing but that need to be confirmed in the analysis phase before they are reported to the team.

2.2 Quarterly Accomplishment #2: Users Interviewed

From May 30 through July 24, 2007, Lisa conducted thirteen interviews with users of scientific distributed computing technology. Five of these interviews were observed by Lee. The individuals interviewed represented eleven projects spanning 24 scientific fields. Eight funding agencies sponsored the projects. Each project is based in the United States, though several include international partnerships.

During the approximately hour-long conversations, the users talked about their work-related goals and challenges. Project affiliations of the users included ALCF, CNARI, ENZO, Lattice QCD, LEAD, LIGO, MEDICUS, MILC, OSG Engagement VO, RadGrid, and TIGRE. One project name has been withheld at the interviewee's request.

Those interviewed were asked to identify their job type. The thirteen interviewees described themselves variously as developer, Ph.D. grad student, professor, project lead, science and technology liaison, scientist, storage engineer, system administrator, system architect, and system designer/developer.

The respondents were also asked to name the scientific field(s) associated with their project. Answers included air quality modeling, astronomy, astrophysics, atmospheric science, bioinformatics, biology and medicine, computer-aided diagnosis, computer science, elementary particle theory, geophysics, gravitational wave astronomy, gravitational wave physics, lattice QCD, materials science, medicine, meteorology, neurology, neurobiology, neuroscience, nuclear engineering, physics, psychology, radiology, and speech pathology.

The funding agencies supporting the respondents' work included the CureSearch National Childhood Cancer Foundation, Department of Energy, National Cancer Institute, National Institutes of Health, National Institute on Deafness and Other Communication Disorders, National Science Foundation, and the State of Texas.

2.3 Quarterly Accomplishment #3: Transcriptions Completed

Nine interviews were transcribed during the reporting period, totaling approximately 47,000 user words and more than 12 hours of interview time. The observations for the reporting period (Section 3) are based on these write-ups.

The interview transcripts will be included in their entirety in the final User Perspectives report. To ensure their accuracy, Lisa sends a copy of the completed transcript to the interviewee for review. All comments to date regarding the quality of the write-ups have been positive; a few tweaks to the answers have been requested.

Two users were asked via email to provide additional information after their interview. In one case the user was asked for more information on a reported technology problem. In the second case clarification on a few of the users' answers was requested. Neither user has supplied the requested information to date, though one has indicated an intention to do so.

Figure 2 shows an overview of the nine users featured in this report. A high-level description follows. Masculine pronouns have been used to help preserve anonymity:

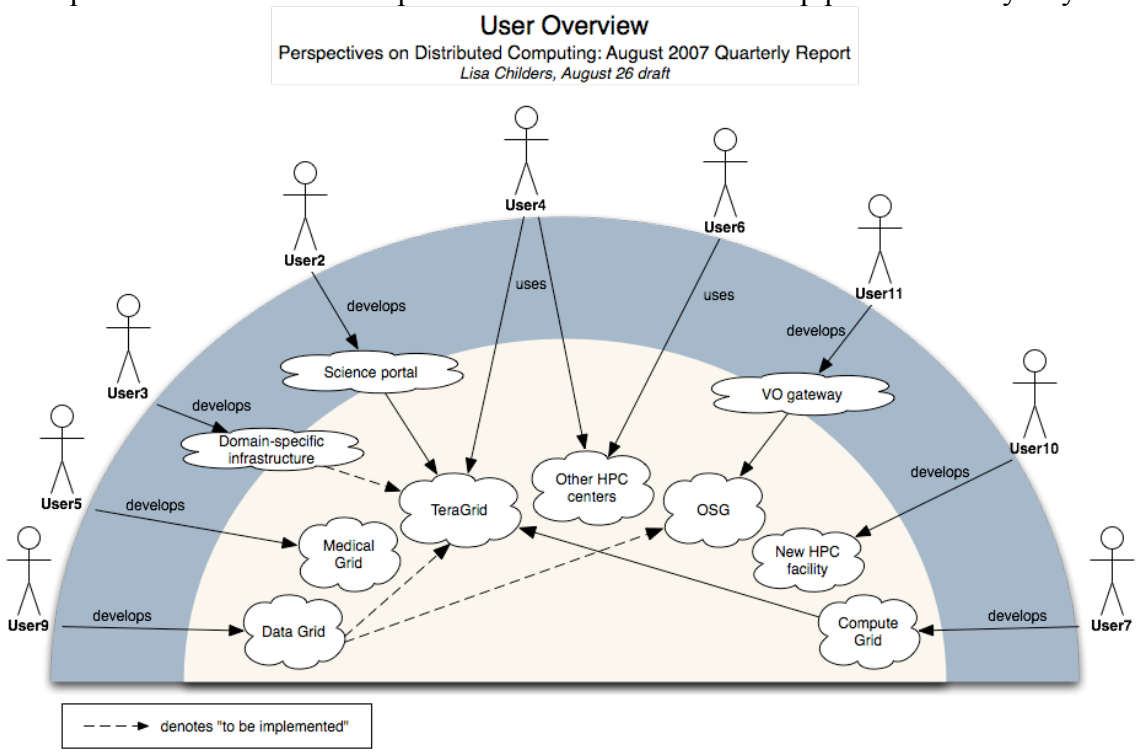

Figure 2: Overview of users covered by this report

2.3.1 Science and Technology Liaison/Portal Developer

May 2007 interview
Featured interview quote: "**Troubleshooting currently requires knowledge about software internals**/"
User2 works with both atmospheric and computer scientists to implement meteorological use cases in a portal environment. *User2* describes himself as simultaneously assuming the role of a CS user and a scientific application developer. His work includes integration and testing of distributed computing tools and clients. The Globus services used are maintained by TeraGrid. Familiar with Grid computing concepts, *User2* provides a seasoned, practical view from the client side.

2.3.2 Scientist/Developer

June 2007 interview
Featured interview quote: "**The Grid is a black box to me**."
User3 is building a framework that enables neuroscientists to store, analyze and share data. The user is not a Grid computing expert, but collaborates with people who are. He depends on these experts to translate his domain-specific needs to the Grid. The group is working toward using TeraGrid resources to scale processing of high-volume domain-specific data.

2.3.3 Developer/Scientist

June 2007 interview
Featured interview quote: "**The reason my tasks are so time-consuming is failure**."
User4 maintains a simulation code that runs at the largest computing scale currently available. His main focus is on the HPC aspects of the simulation as opposed to the science: fixing algorithmic deficiencies to make the code run at the next largest scale. His application is demanding not only in terms of CPU usage but also in terms of producing prodigious amounts of data. Because the application pushes so many system boundaries, *User4* encounters problems not experienced by the "thousands of users chipping away at small jobs."

2.3.4 Developer/Scientist

June 2007 interview
Featured interview quote: "**Performance improved from days to seconds**."
User5 is in the fourth year of building infrastructure enabling radiologists in North America to share medical data in research and clinical settings. His developers create higher-level capabilities by integrating domain-specific code with the programmatic interfaces of Globus services. One of the few open source medical applications in existence today, *User5's* project successes are serving as a catalyst for change in the field of medical technology.

2.3.5 Scientist/Developer/Project Lead

June 2007 interview
Featured interview quote: "**The Grid idea is great, but there are barriers to making it work today**."
User6 is an experienced lattice gauge theorist. He is working to understand the interactions of quarks and gluons and applying that understanding to the discovery of new, fundamental parameters of elementary particles. Looking at Globus for help in managing lattice files, *User6* provides important feedback on the barriers facing new users.

2.3.6 System Architect

June 2007 interview
Featured interview quote: "**If things don't work you need an expert to fix them**."
User7 is building a Grid enabling the State of Texas to provide scientific applications for research and education. A secondary goal of the effort is to help develop the business of the state, in terms of providing greater access to resources and knowledge. *User7's* team provides high-level support, helping users translate their applications to the Grid. *User7* describes his approach for building the Grid and also outlines some of the problems infrastructure providers face building these systems.

2.3.7 System Architect/Scientist

July 2007 interview
Featured interview quote: "**Globus enables more science**."
User9 builds infrastructure for distributing gravitational wave data detected from astrophysical sources to analysts around the world. Having created a data replication Grid that is robust and reliable, *User9* and his team are now working to reduce the time needed to configure, maintain, and manage it. Other work includes enabling production data analyses to run across a greater number of resources, with plans to federate into Grids such as OSG or TeraGrid. *User9* provides a view on the challenges scientists face keeping track of data: not only coming off the sensors, but also tracking results from Grid-facilitated analyses.

2.3.8 Storage Engineer/Project Lead

July 2007 interview
Featured interview quote: "**Solving problems is easy once you have all the data**."
User10 is building the storage and I/O systems for a new leadership-class computing facility, due to come online in late 2007. Many of *User10's* future users run their applications at HPC centers that exist today. The new facility will offer scientists opportunities to scale their work by increasing the size of a mesh or the number of molecules simulated, for example. *User10* describes issues associated with providing data services that will operate at unprecedented scale.

2.3.9 System Designer/Developer

July 2007 interview
Featured interview quote: "**Resource usage within our Virtual Organization is opportunistic**."
User11 builds infrastructure to lower the barrier to entry for the Open Science Grid; a complementary goal is to bring new users onto OSG. *User11's* project does not own any

OSG resources: it provides users access to resources left unused by the major OSG-based experiments. Much of *User11's* work involves detecting and fixing problems with OSG infrastructure before his users encounter them. *User11* provides a glimpse of both the challenges and opportunities afforded by today's distributed computing tools.

2.4 Quarterly Accomplishment #4: Initial Issues Summarized

Based on the interviews transcribed to date, we have constructed initial summaries of user issues, user expressions of satisfaction, and the users and their projects. We note that we have not yet interviewed enough users for these summaries to be quantitative or to be used in prioritization activities. The current summaries merely report the issues that we have observed in interviews so far; there is no relative weighting of issues.

The initial *user satisfaction summary* is useful because it identifies some of the specific aspects of Globus components that users find valuable. Four specific components—GRAM, GridFTP, RLS, Security—are discussed, and for each of these, users reported one or more specific points about their usefulness. Users also made comments about the usefulness of the Globus Toolkit installation procedure and documentation. Perhaps most interesting, users mentioned general aspects of Globus software that they find particularly useful, including the open source approach, backward compatibility with earlier versions, technical support responsiveness, and usefulness in accomplishing science.

The initial *user issues summary* is notable because in addition to a set of issues with specific Globus components, users expressed another set of issues relevant to the use of the software in production-oriented activities. The four general categories (reliability, diagnostics, communication, and technology adoption) all speak to the fact that users are using Globus software in production environments, and that their needs go well beyond basic functionality requirements. We note that none of these requirements would ordinarily be met in computer science research projects because all of them exceed what would be necessary to accomplish basic computer science research. Further, none of these requirements could be satisfied by initial software development projects: they all require a sustained effort by a stable operations and maintenance team after the initial development has been completed.

The initial summary of *users and projects* shows that Globus software is being used in a wide set of environments. While TeraGrid and Open Science Grid are important infrastructures for some users, there are also other HPC centers and domain-specific Grids in which Globus users participate. The summary also shows that the users have diverse roles in their projects. Some are end users of applications based on Globus, and some develop those applications. Others develop or operate systems on which Globus applications run. A few develop mediating services such as portals and gateways. We believe that it is notable that Globus software has a role in all of these diverse activities, suggesting that the requirements addressed by Globus software are fundamental.

2.5 Quarterly Accomplishment #5: Issues Dispatched

User interviews often point to specific issues that specific people are having. In some cases they may represent issues that many users are having. While the greatest benefits of the User Perspectives project will be realized with the documentation and analysis of dozens of user interviews, issues that might be suitable for immediate follow-up are dispatched to appropriate CDIGS team leaders. We do not track the outcome of these issues, as it is expected that the development teams will track them. This section summarizes the issues dispatched for the reporting period.

2.5.1 Number Dispatched by User Project

- 4 issues from Lattice QCD
- 1 issue from ENZO
- 5 issues from OSG Engagement VO
- 3 issues from LEAD
- 2 issues from TIGRE
- 1 issue from MEDICUS

2.5.2 Number Dispatched relating to Globus Technology

- 1 issue for Swift
- 2 issues for RFT
- 2 issues for GridFTP
- 4 issues for Java WS Core
- 7 issues for GRAM

2.5.3 Number Dispatched to Globus Project Representatives

- 2 issues for TeraGrid
- 1 issue for OSG

3 Observational Highlights

This section lists two types of paraphrased observations taken from user transcripts: expressions of user satisfaction and reports of user problems. The excerpts have been edited for flow and readability.

3.1 Excerpts of Reported User Satisfaction

Figure 3 shows an overview of the reports of user satisfaction for the initial period. Individual observations follow the figure.

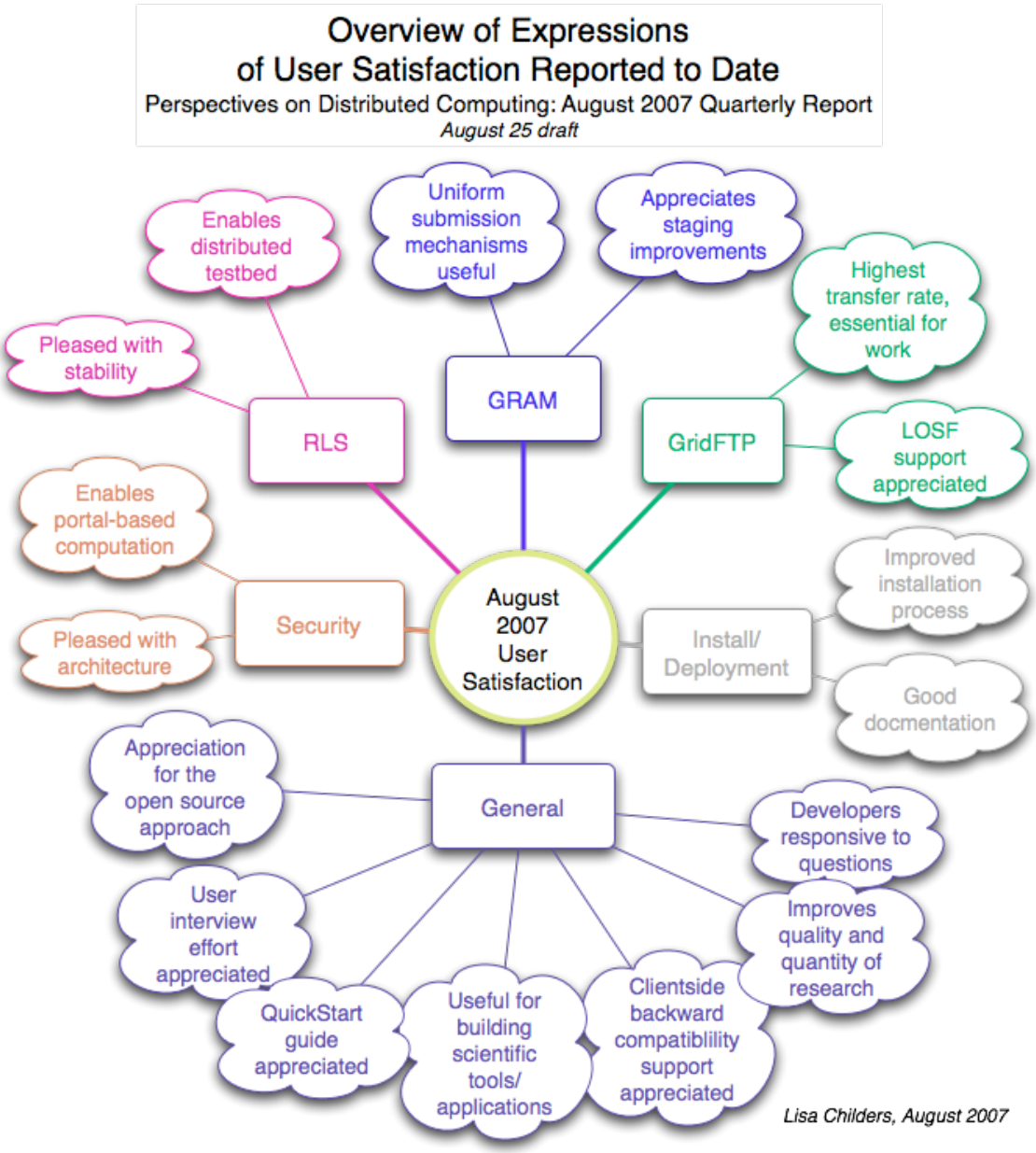

Figure 3: Expressions of user satisfaction reported to date

3.1.1 General

- I really appreciate the overall effort. For Grid the whole paradigm can only thrive is if there's an open source and standards-based implementation, and the Globus Toolkit is delivering exactly that. One problem in the medical domain is that the products of software and hardware vendors are proprietary. The Globus Toolkit and the Grid paradigm can have a major impact on how medicine is being addressed from the technology side.

- It's really not a lot of work to customize the Globus tools to fit their own users' use cases. I see TeraGrid doing that, and I think that's great.

- I appreciate having people who build the software actually look at how people are using them. So this interview process is useful.

- Client-side backward compatibility is important. When a new version comes out I should not have to rewrite my software. This has been a major concern in the past but has been much better lately. If the clients can talk to the services in the same way and get the same functionality, that is good.

- With the Globus team I generally feel like when I ask questions, they're quite responsive.

- With the Grid technology we deliver images from any clinical trial center into the radiologist's own review workstation. That's possible today using Grid technology, and it's quite rewarding to see. The radiologists are very pleased. This actually engages more radiologists in clinical trials than before, so we are improving the quality and quantity of research being done.

- For someone like myself who is not a computer science person, the WSRF framework allows me to quickly and easily use things like resource properties. Lifetime management, subscription and notification – all nice things. It allows me to use them quickly and much more easily than I could do if I had to code all that stuff by myself. After all, I am a physicist, and I'm dangerous when I'm writing code. The more code other people write for me, the better.

- The QuickStart guide is very good, very straightforward and clear. Even for somebody who is a beginner, this is a straightforward document.

- Much of the basic functionality, such as data transport, is included in the Globus Toolkit. And there was not a lot of effort required to make Grid technology work for the medical domain. And that was very neat because there's no need to reinvent the wheel. ... There's no other technology that provides compute resources, data management resources and security at the level that the Globus Toolkit does. Period.

- I certainly appreciate all the Globus team's efforts. The Globus Toolkit has really succeeded in what I think is one of its primary missions: enabling more science. Without a doubt, Globus has made more science possible. Period.

3.1.2 GRAM

- The GRAM interface is really cool and provides us with one common interface for all the job managers. This is important for us since we are required to use multiple clusters and each has its different job managers. GRAM enables us to use uniform mechanisms for both job submission and monitoring. It also supports the authentication mechanism that we like.

- I like GRAM4 because the staging support is better. It's a pretty big improvement over GRAM2 in that sense, because you can do smarter staging (like the whole filelist). That maps much better to our Condor description files. And in general the architecture is better.

3.1.3 GridFTP

- GridFTP exposes the highest rate of network transfer from filesystem to filesystem across a transcontinental distance. For example, if I have to move output from Pittsburgh to San Diego, GridFTP buys me a factor of two or three over other options, which is essential.

- Our major challenge with GridFTP has been the "Lots of Small Files" problem, but it looks to be addressed and we're very excited about leveraging that functionality. Other than that, GridFTP is completely reliable and moves tons of data. What else could I want?

3.1.4 RLS

- The Replica Location Service enables our testbed to be distributed across multiple locations. The replicas also allow us to choose the fastest available compute server for our computations.

- We rely quite heavily on RLS. I think we run the largest RLS network in the world. When RLS goes down, you better believe we know it, as we have to jump into action. Fortunately it very rarely goes down now. We're quite pleased.

3.1.5 Security

- The Grid security infrastructure is one of the things we completely rely on and are building our tools on. We absolutely love GSI-based authentication. It has been really helpful and paved the way for portal-based computation.

- The Globus GSI and everything that's built around that ecosystem now works really well for us. We can hook into it in so many different ways. We can set up services that manage the delegation for the users. The only thing the users have to

do is enter the system once using something like MyProxy. Everything else is handled for them. That works really well.

- Using Grid security you can audit, verify certificates, verify attributes, etc. These mechanisms are superior to current clinical practice, because many of the clinical documents today are written on paper, stored in physical files, sometimes reports are thrown in the trash, etc. So there are many places in the current system where private information is exposed to the outside. This is a way Grid technology can help, because it has a very good security model.

3.1.6 Install/Deployment

- The Globus installation has improved a lot. I've been installing Globus from the first version to today, and today's installation is very good. Also the problems we encounter during installation are documented very well.

3.2 *Highlights of User Issues to Date, Categorized by Theme*

Figure 4 shows an overview of the user issues observed during the reporting period. Individual observations follow the figure.

Figure 4: User issues reported to date

3.2.1 Reliability Issues

Two types of reliability concerns have been expressed in user interviews thus far, both of which suggest that additional fault tolerance and resiliency improvements would be well received. The first type of concern is expressed as **system-wide failure**.

- At 60,000 processors (or whatever it's going to be) computation at that scale will not be possible using the current approach to batch production. How will a user be assured when their timeslot comes up that every single component of the system is functional? And how long will it stay in that state, given the mean time between failures is proportional to the component count?

- I'm running in the 2,000–4,000 CPU range at the moment, and within the next year that will increase to the 32,000 CPU range (perhaps even a factor of two more than that.) However, nothing really works reliably today at the 2,000- or 4,000-processor level; I'm referring to filesystems, batch-process launching systems, disks, monitoring tools, etc. Thus I am doubtful about these things reliably working at a level ten times greater than that.

- One idea for mitigating the effects of system failures for the type of job that uses an entire computer system in a single run: move away from the batch-queuing model. Move to a support model that is closer to a physical experiment, such as with an astronomical telescope. Provide the ability to schedule the run when systems staff will be on call to fix problems immediately.

- It is much more common to suffer a failure in the first few seconds of a job run, as opposed to the last seconds. If a node, for example, can't see the parallel filesystem, that might be fixed quite quickly by a sysadmin. But if you're running in batch mode and have waited days (if not weeks) till your batch job starts, it might instantly fail and then you have to go through the whole thing again. So the operation of these scheduled runs needs to be made more reliable in both physical and human terms. You've got to have adequate systems support to overcome problems in real time.

- We have so many things to keep track of we're losing the ability to keep track of it. We're building tools that need the information to help the users accomplish their work. But the systems that provide the information are breaking underneath the load. Then all these tools and infrastructure become unworkable and work stops.

- The difficulty is not that things break; the difficulty is in detecting that something is broken. I may not even know who owns the site—it's just a black box to me. So I have to figure out what went wrong. I do a little bit of probing, and then either tell the remote site what went wrong, or fix my stuff. Most of the times it is easy for me to figure out what is going wrong once it is detected.

- Stability for me in the context of filesystems means running without failures: fileservers don't hang, user jobs run to completion, etc. Right now our filesystem hangs and jobs have to stop because they can't write data. We've got to get to the point that something figures that out, a backup comes into play, and the job can continue.

The second type of concern is expressed as reliability at the **service level**.

- There are maturity issues with some of the tools: they are buggy and some capabilities don't work as promised.

- I would like to have many of the basic services like GridFTP and GRAM made more reliable before I see additional features coming out.

- From a workflow point of view, the most time-consuming aspect of data transfer is dealing with failures. It takes as much human time to deal with the one file that didn't move successfully as with the ninety-nine transfers that succeeded. Human intervention to deal with failure is expensive.

- A job can fail in multiple ways. One way is that the job is successfully submitted to a site, but something fails during the job run. This can happen when, for instance, a filesystem goes away while the code is running. To detect that type of failure is not easy. The error codes returned for this type of problem vary; some schedulers will say the job was successful while others will say that it was not successful.

- GRAM2 has been more stable and reliable on TeraGrid than GRAM4. That is the only reason I prefer GRAM2 over GRAM4. I need at least 70% success rate to consider a service stable. Ideally we want it to be much higher, but with GRAM4 we are seeing a lower success rate. I certainly don't want to blame everything on GRAM4. We've seen hardware failures on the cluster side. But GRAM4 should improve the way it responds to hardware and network failure.

- My goal for my data replication network is to have a mean time between failures of three months.

3.2.2 Diagnostics

In user interviews conducted thus far, observations related to diagnostics fall into two categories. The first is interpreting **error messages**.

- Error messages are very cryptic and cannot be parsed in a way that would enable us to build automated adaptive behavior. The most common error we see is "login incorrect," yet the source of the problem is usually not a login issue. The error message can be triggered by a hardware problem (e.g., a node is down in a striped GridFTP deployment), or an allocation is out of the limit, or the scheduler might be paused, etc.

- There is a lack of documentation about errors. For example with GRAM, all we get is an error code and there is not enough documentation explaining the error. We then have to google to find out how other users handled the problem. Sometimes we even need to go as far as to dig into the GRAM source code to determine under what conditions the error is sent. There is some error code documentation, but not enough to be useful.

- Sometimes we get weird errors using GRAM2 that don't really reflect what's going on. Weird errors like "Error code 17" that supposedly means one thing, but is most commonly due to something else. For example, it says, "Could not create job description file" when the real reason is the user doesn't exist.

- When middleware cannot determine the particular error (and it's reasonable that it cannot determine everything), I would rather it propagate the original error message. Send it up the middleware layers of the architecture, instead of misinterpreting something and issuing a misleading error message.

- If we see certain errors right up front, we cannot directly take that and send it to, for instance, the TeraGrid helpdesk. I have to do at least an hour of digging. Because if I send directly an error message to the helpdesk they will reply, "This is something to do with your client side. There is something wrong." So I dig deeper and deeper and go through my usual tests, and see, "Oh this service is down. Ok, here is what's happening."

The second category of observations related to diagnostics involves **troubleshooting**. Users indicate a lack confidence in dealing with runtime problems. The statement that troubleshooting currently requires a deep understanding of implementation details is particularly noteworthy.

- Finding out what to do next or troubleshooting is not something I am capable of doing – not at this point, without going through a learning process, which I don't have time to do. Let's say I do a globus-url-copy from one center to another, and I get an error message saying "End of file encountered," and the file at the other

end is of zero length. Now what do I do? Right now, I send an email to the administrator asking, "What does this mean? Why didn't it work? It worked six months ago."

- When anything goes wrong with your certificates, site certificates—anything like that—it's beyond the scope of anything a user can deal with. And usually it's beyond the scope of what the computer center personnel can deal with as well. It usually means that you're just crippled for a couple of days until the one guru at site X can actually figure out why what used to work no longer does.

- There is a lack of documentation is in the troubleshooting area. System administrators struggle when a problem is happening, and they need to figure out how to solve it. Troubleshooting right now requires knowledge about the internals of the software, so if expertise is missing on the admin side, the problem can sit unresolved for three or four days.

- Solving problems is easy once you have all the data in front of you. The hard part is getting the data and knowing what data to get. Networks are black boxes. Very rarely do you have access to operational status on routers and the like. So you must infer what's happening using tools like Iperf, netperf, pipechar, etc. I would love to see the network become a white box so it could be inspected like a scheduler: current status, load, expected queue time, etc. Then you could write software that could do optimizations and decide which routes to take.

3.2.3 Communication Issues

Four types of communication issues have been observed thus far. The first type of issue has to do with **how Globus developers communicate** with the community.

- I think there's a reliance on the email lists for archiving information. Archives are great, because sometimes the details really only exist in an email list and you want to be able to find them. But it can be hard because there are so many email lists to monitor.

- It would be helpful if some of the workplans were documented in a more centralized place. Information exchanged through the email lists pertinent to the roadmap or the campaigns could also be recorded in the same centralized place.

- We're trying to be deliberate about design and think ahead in terms of how the pieces of the system should fit together. We would like to understand what the tools will look like in one year, two years—even three years from now: how RLS, RFT, etc. are evolving, and what the priorities are.

- The Globus team has gotten better at this, but there are still times where the team appears to be self-focused or focused inward. This doesn't apply across the whole team. But some folks seem to be focused on infrastructure for infrastructure's

sake, as opposed to infrastructure for other people to build on. There are still some pockets of that occasionally. But that's certainly not the rule. As I think about it, the teams I've interacted closely with—the RLS and GridFTP teams—I can say that's the opposite. They tend to be very supportive in terms of reaching out, asking for use case scenarios and requirements, and being responsive to input.

The second type of issue reported by the user is a sense that there **aren't general opportunities for scientific users to express feedback** about the systems they use:

- The way the HPC centers work, all the information comes down and there's no feedback, this conversation not withstanding, from the poor users of the system who are forced to use poorly designed and inadequately supported computers. We suffer terribly in loss of scientific productivity dealing with the endless failures at every level of these systems.

- There is no feedback from the users to HPC center management or the NSF, in terms of the cost in human resources of using their systems. The current round of the NSF program is a perfect example—an obsession with buying a petaflop computer, with
    - No input whatsoever from the userbase;
    - No clear understanding of how it possibly could be used;
    - No input from the end-users as to its architecture, its characteristics, or what it will support.

The third type of communication-related issue involves **perceived differences between the user's worldview and that of computer scientists**.

- In some areas of science people use GUIs and hit a button to get answers. But that view of computing just doesn't work when computation time is measured in months. For example, I started working on my current simulation six months ago and it hasn't completed yet. My work doesn't fit well with the computer science idea of running myriad little processes.

- Just because it's old doesn't mean it should be ignored. Ninety percent of the science codes in a recent Oak Ridge survey were found to be written in Fortran, for example. No one in the computer science community can be bothered to help a Fortran programmer anymore. They probably don't even know Fortran. But in science it's still tremendously important, and C is the next one behind that. We're not going to switch languages. I'm sorry to say that the DARPA HPCS language initiative is pie-in-the-sky. As somebody who remembers Ada – this is even less likely to work.

The fourth type of issue is related to **intra-project communication**.

- In a project as distributed as mine is, you spend a lot of time in meetings and writing emails, just communicating issues back and forth. There are so many sites, there are so many projects and experiments and they all have slightly different agendas. So something that might be important to you, nobody else might care about (or the other way around.) Or you get pushback from people on something that doesn't make sense to you.

3.2.4 Technology Adoption

In user interviews conducted thus far, issues related to technology adoption fall into four categories. The first category of technology adoption issues pertains to difficulties **using deployed national scientific computing infrastructure**:

- GRAM4 is supported on a low-priority basis on OSG. GRAM2 is a production service for OSG, but GRAM4 is not. So if GRAM2 fails for them, the site is considered to be failing. If GRAM4 is failing, it is not that big of a deal.

- There are no obviously robust methods that we use to help us get around failures during data transfer. There is a thing called Reliable File Transfer (RFT) that might help, but we don't just use NSF TeraGrid; we use DOE centers as well. It's not clear to me that they would implement anything like that.

- Security requires Byzantine communication between centers. For example, try using an NSF certificate at a DOE site. Dead on arrival.

- I was trying to do a file transfer from NCSA to a special tape archive at Fermilab that's managed by dCache. To make this work I needed to use SRM-copy. And the SRM-copy was failing because Fermilab has to set up certain map files to make that work, and they are not being properly maintained. So I finally just fell back to the FTP again. But in order to get the files onto the tape archive at Fermilab the scp has to go through two stages: moving the files from disk to disk, then from disk to tape. It's a painful process and doubles the amount of work.

- Once I get a transfer running right, it's pretty much fixed—that is, until the next time the admins do a kernel upgrade and blow away all the configuration modifications I made. This type of thing happens frequently.

- Recently we've had problems with sluggishness on some of the networks like the ESNET. The file transmission rates are painfully slow, errors occu, and then we have to retransmit.

- The users of our Grid are required to negotiate separately with each site for use of cluster resources. So our users need to contact the person at each university who has the power to authorize them. The authorization process is not always easy

because many of the universities are organized to serve local campus users. It's easier when the universities are TeraGrid sites. They can say, "Sure we'll give you a little starter account, and you can go through TeraGrid to get more cycles."

- We don't provide a standard system service that could be considered a metadata service, whereas we do provide standard system services for I/O, data movement, archiving, etc. The reason for this is because to date, metadata has not been solvable in a general way. Metadata very quickly becomes very application specific. And though most scientists have perhaps not as a good a system as they would like, they do have a system of some kind that they already use for tracking metadata.

- When moving data in and out, our facility only controls one end of a GridFTP transfer. We can make sure the machines on our end are beefy enough and are configured correctly and tuned right. But if the guy is trying to transfer the other end off his laptop, we'll only go as fast as his laptop. Data transfer is an interesting problem in that respect. It's a two-ended problem. If you're trying to schedule the transfer, it requires co-scheduling—you must schedule resources at both ends. You don't have control over your own destiny: you can control your end and you can coach the other end. But if one end doesn't have the hardware there's nothing you can do. And that actually gets quite frustrating. While I know that rationally they understand it, all the user knows is he's not getting what he wants.

The second set of adoption issues concerns the **perceived overhead of using GSI**.

- GridFTP carries all the baggage of Globus with it but it is the only component we're interested in. Really it's just an FTP program—why on earth do we have to bother with all the certificates and all the stuff that goes with it? All we want is fast and reliable point-to-point transfer.

- Just getting Grid certificates requires a lot of knowledge:
  - Knowing which one is the best one to get
  - Knowing how you use those certificates to authenticate
  - If you've gotten one from somewhere, how you get to another place and get authenticated there

- I would not consider installing Globus myself because of the overhead. When things go wrong with certificates it is beyond the scope of what a user can fix.

- Running a CA, deciding whom you trust—that's all a large pain. For example, you have to get a CA certified by TAGPMA and buy special hardware. And after all this is done, as a user you don't gain much of anything—no additional capabilities—you access the same machines as you could before. It's a big hassle for some potential benefits (like delegation, having your own agents out there to

do things for you, etc.). There is some potential there, but it hasn't come to pass that we've needed it.

- If I could be convinced that a non-GSI version of GridFTP was stable and secure, I'd use it. I hate GSI. It's very good at what it does, but it is a pain. When I was involved in GridFTP development, GridFTP didn't have problems—GSI had problems. Once I could get people past the GSI issues and get it all configured, GridFTP just runs. But the big thing that GSI gives you that no other solution does is delegation. GridFTP particularly needs delegation because it does third-party transfers.

The third category contains observations about **various barriers to entry** for people who are not Globus experts.

- It seems to me that in order to get Grid solutions you have to be pretty tech savvy. Getting the certificates, doing the job submission, doing the DAG of the workflows on Condor, managing the security: all of that seems to be an enormous barrier for actually getting jobs done. I do not know of a general mechanism that scientists can use to communicate need scenarios and work out how to solve problems together. My project is lucky to have experts on staff who translate our needs to the Grid.

- The VDT is very helpful as far as getting things deployed much easier than in the past. But then after that, trying to get users working with those deployed tools is still a problem and takes a lot of our time to help users. Ease of use is still a big problem.

- The complexity and reliability of the tools we have to work with are a key problem for us. If you add up all the available Grid tools you don't get a very good user environment. It is too difficult for most users. For example, there's only been one person I've worked with so far who can really figure out on his own how to use the Grid tools we've deployed; he's really an exceptional kind of person this way.

- It would be tough for someone unfamiliar with Grid computing to build the Grid I am working on. A capability like metascheduling, for example, is not available in a well-known distro (like VDT) because consensus hasn't yet been reached on which metascheduler is the best. Because I am familiar with the area I know who to contact about metascheduling and can work with them directly. Compare this situation to something like Linux, where the novice can find everything he needs in one distro.

- The management of credentials directly is too difficult for our users, so we are moving away from that approach. We plan to start relying on MyProxy and similar types of repositories so users don't have to manage their PKI credentials themselves.

- There are sociological barriers to sharing data in the medical domain, as well as legal issues regarding patient privacy. Once these two issues are addressed, it will be possible to aggregate interesting information on the Grid.

- If we can get a lot of users on our clusters with minimal interaction with user support staff, we should be able to do as well when getting them on the Grid.

- There are still not enough people in the world that have real in-depth Globus knowledge. Certainly they're hard to find and hire. So we train people here, to the best that we can. But it's still hard to say to someone: "I'm thinking about putting RFT into this service, but need to understand where it's going to break. Stand up RFT and throw larger and larger requests at it until it breaks." I usually end up needing to kick-start the effort and spending a more management time than is optimal. That is not a comment on my staff because they're all good, hardworking, smart people. They just don't have some of the expertise, especially with of the Web services stuff now in Globus Toolkit 4.

- People write great developer guides. Those are great for somebody who is a developer. But what about the rest of us? Non-experts sometimes don't understand the big picture – concepts as simple as "What is a client? What is a server? What's a third-party transfer?"

- I would also find more tutorial-like information quite useful. For example I read the whole Globus Toolkit 4: Programming Java Services book and I practiced a lot of examples in there. This is kind of helpful, and we would like more examples. Like when we are writing clients to a GRAM service we look for more tutorials or even CoG help in some sense. More tutorials would be helpful.

The fourth category includes observations relating to the **integration of new technology**.

- "Keep it simple" is the advice I would give to people building systems for use by scientists like me. If we scientists have any spare time or any spare brain cells, we want to spend it on adding sophistication to our own domain-specific code. For us all the complexity should lie in writing our own code to solve systems of equations that describe whatever it is we're trying to do.

- I look at how easy it will be to pull the pieces together and build a higher-level functionality that meets the needs of the users I support. So I might look at a tool that purports to do something, for example "manage data transfers" or "manage

workflows." I then try to decide if the tool is one that, while offering bright shiny new functionality, is at the same time unstable and unreliable.

- When evaluating a potential service, I look at its interfaces. I need to glue all the pieces of my system together in reasonable ways, so I first assess how easy it would be to add the component into my system. If the API language is not my first choice, then perhaps my team would be forced to extend outside its area of expertise. If it is API agnostic, then I can easily just write whatever I want.

- What logging features does it include? Will I be able to drill down easily when I need to identify the source of problems? Can I turn up the logging levels to get a picture of what's going on?

- An important consideration is whether or not a tool is extensible and allows me to build on top of it. This is in contrast to tools that try to provide a complete solution that force me to rip and replace stuff. I stay away from such tools because they require me to give up other stuff that I'm already doing in order to use them.

- Most of our applications don't call any security APIs. They need to have the environment and security managed for them. So I can't go to a data analyst on our project and say, "I need you to link with this library so your tools will interact properly with the security." They expect the infrastructure to operate at a level either above or below that, depending on how you characterize it. They just want to run their job, and they want everything to be handled for them.

- The attempts from Globus-related teams (I don't think these are Globus Toolkit proper) to provide tools and infrastructure to help with provenance have required too many application level changes. The approach was, "Just do everything this way; then you'll get the provenance information." But there's no way to "just do it this way." That's not the way my users can be approached. The users are going to do their science. The science is going to lead, and all the other stuff has to be tacked on.

3.2.5 Other Technology Issues

We collect here a number of miscellaneous comments about technology issues.

Issues relating to **C WS Core**:

- More C WS Core code examples and documentation are needed.

Issues relating to **GRAM**:

- A concern for GRAM4 on TeraGrid is the container goes into hibernation for a while without any explanation or log messages, and comes back by itself after a few hours.

- Cluster users often modify their environment settings for their jobs. They want those values to be used for the job, but GRAM4 ignores them. In contrast, if users submit the job directly to the scheduler, those values will be honored.

- Many defined RSL attributes are not implemented in the backend scripts. For example, the default LSF.pm scripts did not include support for taking the min memory XML-based RSL attribute and turning it into the right LSF line in the submit script.

Issues relating to **GridFTP**:

- We have a problem using GridFTP with firewalls and active/passive settings. Different combinations of active/passive settings are required depending on the host pairs. For instance, for some host pairs we need to make the source active and the destination passive, but for others we need to make both active. So we've had to do all sorts of hacks to switch settings at runtime.

- The interface to GridFTP is a bit clunky—we would like something as simple as scp. I gather that the TeraGrid project has done a fairly good job encapsulating some of the knowledge you need into tools such as tgcp, but I get the impression that some of those things aren't well-maintained.

- I tried to install the GridFTP client myself, but it failed on Solaris, and then I gave up because I could use it from Fermilab. I didn't try to track it down further, but when I was trying to install it, it looked like it was trying to pull half of the Internet onto my workstation. Part of the problem, I think, was that I ran out of disk space.

- GridFTP docs should include an engineering guide written for sysadmins (or people about to install a GridFTP server). The document would walk you through the configuration considerations and help you prepare information needed at install time:
    - How big does the machine need to be?
    - How big do the drives need to be? How fast?
    - What should the network connectivity look like?
    - Should I run a striped server? Should I not run a striped server?
    - Should I run GSI?

Issues relating to **information systems**:

- Based on practical experience, the tools monitoring GRAM and GridFTP do not show whether a service is really available. If you ping GRAM or GridFTP it works fine, but sometimes when you try to do a functional thing (like transfer a file or submit a job), it fails. So instead of using INCA-based tools or WebMDS to find out if a service is available, we do a test run (execute a /bin/date or transfer a file) prior to the actual request.

- The MDS4 Index breaks, it's slow, and I find it to be overly complex:
    - XML and XPath is more than I need 98% of the time.
    - Java makes it quite heavyweight for small things.
    - The last I heard they were running in memory instead of out of a disk-based database, which hogs a lot of memory.

- The number of data products we are responsible for is growing quickly, so the number of files is growing quickly. For us the big issue is not so much the amount of raw data, because it's still a terabyte a day. But now it's divided over tens of thousands of files per day instead of being spread over a couple thousand files. And they're all different sizes. We now have so much information to track about our data we're getting killed by the metadata.

- Information we need to know about a resource in order to determine if it is suitable for our work includes the following:
    - Can I build my code here?
    - Will my problem fit on this machine?
    - What is the operating system?
    - What is the software that's already been installed?
    - Related but different: Is my prerequisite software installed?
    - The number of CPUs
    - The number of nodes
    - The amount of memory
    - The disk quotas
    - The scratch disk space

- If I can find all my bank history in a split second, why can't I find a machine that meets my requirements in less than ten seconds?

Issues relating to **Java WS Core**:

- The major problem with the Java container is in the area of database connectivity. Compiling the container can be very simple or painful, depending on whether or not you need ODBC drivers.

- The default use of PostgreSQL in the toolkit should be reconsidered. MySQL is more common than Postgres.

- The Globus MySQL installation instructions and the way the database is connected should be reworked. You have to install a specific version of the driver, and some of the drivers you can't get anymore because they're outdated.

- More dynamic IP address handling is needed. I'm referring to the way the container handles the network coming and going. The use case I'm dealing with is where there's a sensor somewhere connected by a GPRS cell phone. The sensor gets different IP addresses every time it connects. It's just up for a few minutes

and then goes down again. The current notification framework doesn't really work well in that dynamic scenario.

- I would like the delivery of notifications to be guaranteed. Let's say we have a sensor that needs to aggregate some data. So every now and then it pops up and says, "Ok, here's my data for the past hour," and sends the data to a service. At the same time it would check for any pending updates from the service, so it would process notifications sent by the service while the sensor was offline.

Issues relating to **RLS**:

- As it turns out, relational databases are not the best way to model our data. We don't really use the relational aspects of it. What we really want are fast index hashes. So I've asked the RLS developers to think about abstracting RLS so it can support other plug-in backends, just like the GridFTP supports other data storage interfaces (DSIs). I would like RLS to support different DSIs. It should have the relational database as the default, but also provide the option of using other methods of representing user data and its mappings between logical and physical filenames. I really like the RLS API and I like the model. I'm very happy with it as a service at that layer. What I want to get away from is the relational database backend because I don't think it's going to scale for us going forward five years from now.

Issues relating to **general install/deployment**:

- I prefer the Linux distro model (or cygwin on windows, or fink on the mac). I can go and pick out what I want, and it almost always works. You get a menu, you pick, it installs it, and everything works.

Appendix A: Overview of the User Advocacy Program

Figure 5 summarizes our current User Advocacy near-term goals. The remainder of the Appendix discusses the motivation for the user Advocacy program, outlines additional goals, and lists the accomplishments to date.

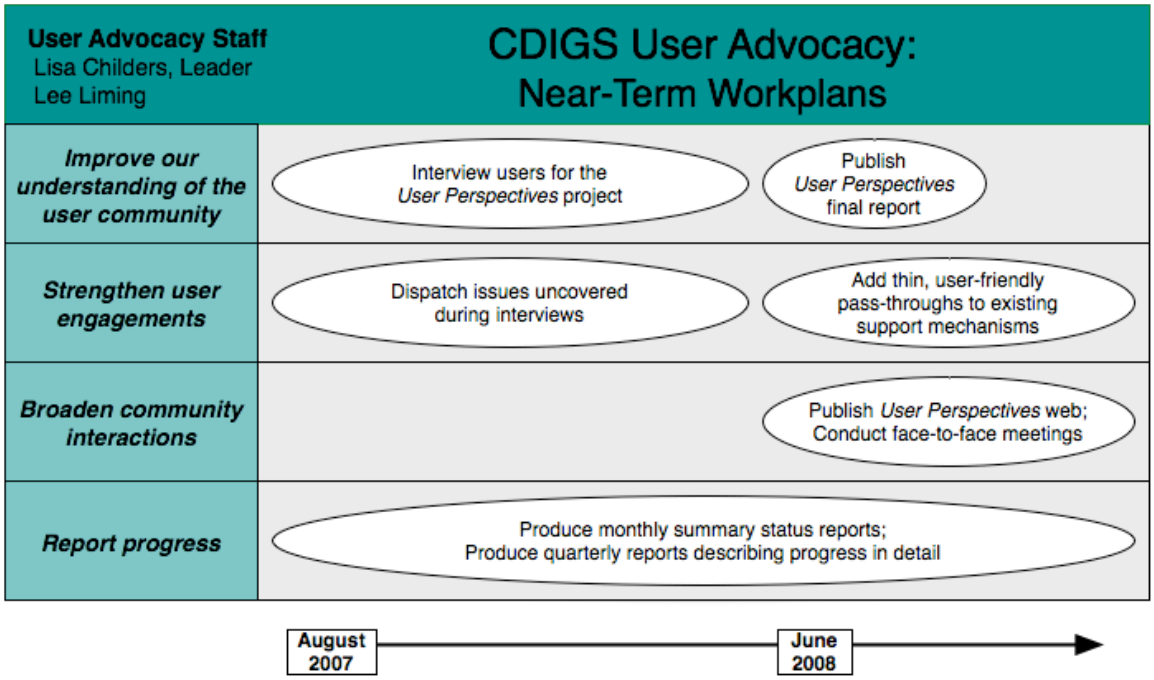

Figure 5: Current User Advocacy workplan

A.1 Background

A.1.1 Motivation

Users of scientific distributed computing technology face many challenges today. Harried middleware developers are pressured to deliver and support lightly tested, sparsely documented features. Users are pressured to quickly get things running in order to accomplish their own work. Everyone is busy, and the software systems being built are as complex as they are powerful.

Given these circumstances it is sometimes difficult for users to efficiently get tech support, in part because it is not always clear how their issues map onto existing tech support systems. While the dev.globus community forum is a powerful and rich tool for interacting with Globus development, it is implementation-focused. Though appropriate for developers, those unfamiliar with Globus implementation details might find it daunting.

The overriding purpose of the User Advocacy effort is to provide the CDIGS team with additional information about current and potential users so its work can be made more accessible to scientific users. Armed with information provided by the User Advocacy effort, the CDIGS team will be more effective in tuning its tech support interfaces, educational materials, and development priorities.

A.1.2 Relation to Current Work

Outreach

Outreach is an activity focused on disseminating information about the group's current and future work. Coordinated by Jennifer Schopf, the Outreach team is responsible for informing the community about Globus technologies, development activities and generally advocating for "The Globus Way." There is no question that the current Outreach work is essential to our success as a group, but it has a different focus from that of the User Advocacy program. User Advocacy is user-centric. The overriding goal of the User Advocacy effort is to better understand and represent the users' view within the team, without allegiance to any particular technology. As such, User Advocacy seeks to complement the Outreach effort.

Project Coordination

We do have user advocacy-like functions in the group right now, such as the targeted user engagements with ESG. But given our CDIGS mandate, we need to extend the reach of these activities. The traditional Globus project coordination model of assigning a representative to each project will not scale to the degree needed. Over time the User Advocacy program will represent the broad interests of a variety of scientific projects. As such, User Advocacy seeks to augment current project advocacy efforts.

Community Forum

As mentioned earlier, navigating the technology-centric organization of our current support infrastructure can be confusing for the naïve. User Advocacy deliverables in the customer support area will involve adding thin, user-friendly pass-throughs into existing support mechanisms. The exact design and approach for this work are not yet known. However, it is safe to say that User Advocacy seeks not to replace or duplicate existing support mechanisms but to provide new paths into them.

A.1.3 Staffing

The CDIGS User Advocacy program is led by Lisa Childers, with Lee Liming participating as available.

A.2 User Advocacy Workplan

A.2.1 Improve Our Understanding of the User Community

Before any real change can take place, the User Advocacy staff must investigate the concerns and problems facing users today. The cornerstone of this investigation is the recently initiated User Perspectives project.

The User Perspectives project will produce a view of users' goals, methods and problems, collected and documented in a methodical and rigorous fashion. Key deliverables include the following:

- Early draft of the User Perspectives report framework (June 2007)
- Additional interviews each month (July 2007–April 2008).The objectives are threefold:
    - Identify individual issues requiring follow-up
    - Forge connections and create goodwill among current and potential users
    - Gather ideas for improving our software, tutorials, and documentation
- Creation of the Perspectives on Distributed Computing final report (June 2008)

A.2.2 Strengthen User Engagements

Over the coming year, the User Perspective interviews will also be used as an opportunity to deepen our engagement with a variety of users with whom we might otherwise not directly interact. Initial deliverables on this front include the following:

- Bug reports and follow-up on individual issues as needed
- Identification of new collaboration opportunities

A.2.3 Broaden Community Interactions

In time the User Advocacy effort will add a new dimension to Globus community interactions. User Advocacy staff will work to build a crosscutting, user-focused community. Deliverables include the following:

- User Perspectives companion web
- Face-to-face meetings tbd

A.2.4 Report Progress

Interim reports will be generated to provide a view on the project's progress:

- Monthly CDIGS status reports
- Monthly bundles of raw interview data
- Quarterly reports summarizing interim findings of the User Perspectives project and User Advocacy accomplishments (August 2007, November 2007, March 2008, June 2008)

A.3 User Advocacy Accomplishments: May–July 2007

A.3.1 Improve Our Understanding of the User Community

- Founded the User Perspectives Project (Childers, May 2007)
    - Designed interview questions (Childers & Liming, May 2007)
    - Conducted test interview (Childers, May 2007)
    - Designed the User Perspectives report framework (Childers, May–June 2007)
    - Reviewed approach with HCI expert (Liming, June 2007)
- Conducted thirteen user interviews (Childers, May–July 2007)
    - Observed five user interviews (Liming, May-July 2007)
    - Transcribed nine user interviews, totaling approximately 13 interview hours and 47,000 user words (Childers, May–July 2007)

- Wrote Perspectives on Distributed Computing: August 2007 Quarterly Report (Childers & Liming, August 2007)

A.3.2 Strengthen User Engagements

- Sent follow-up email to QCD scientist requesting more information on GridFTP transfer problem on June 8.
- Sent sixteen user issues to CDIGS staff members for near-term follow-up

A.3.3 Broaden Community Interactions

No significant accomplishments during the reporting period

A.3.4 Report Progress

- May 21, 2007: Transcription of demonstration interview sent to Fraser, Foster, Liming
- June 15, 2007: CDIGS monthly status reported
- June 18, 2007: Childers, Foster, Fraser, and Liming reviewed User Advocacy proposal and three interview transcriptions
- July 2, 2007: Five interview transcriptions sent to team members
- August 18, 2007: CDIGS monthly status reported
- August 28, 2007: Slides created for CDIGS review
- August 29, 2007: First nine interview transcriptions sent to team members
- August 29, 2007: Quarterly report sent to CDIGS team